

AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A method comprising:

executing a first instruction in a processor;

if the execution of the first instruction generates a cache miss, associating the first instruction with the cache miss;

associating the cache miss with a second instruction dependent on the first instruction;

enqueueing the first instruction for re-execution; ~~and~~
~~enqueueing the second instruction for execution; and~~

after the cache miss with which the first instruction is associated is serviced, re-executing the first instruction ~~and executing the second instruction.~~
2. (Canceled)
3. (Original) The method of claim 1, further comprising assigning an identifier to the cache miss.
4. (Original) The method of claim 1, further comprising determining a priority of the instruction.
5. (Currently Amended) A processor comprising: a re-scheduler to hold instructions enqueueued for execution;

association logic to form an association between a cache miss and ~~an~~ a first instruction generating the cache miss, the first instruction to be enqueued in the re-scheduler; and

propagation logic to propagate the association to ~~an~~ a second instruction dependent on the first instruction ~~generating the cache miss~~, the second instruction to be enqueued in the re-scheduler.

6. (Original) The processor of claim 5, wherein the re-scheduler is further coupled to priority logic to determine a priority of instructions in the re-scheduler.

7. (Original) The processor of claim 5, wherein the association logic is to assign an identifier to the cache miss.

8. (Original) The processor of claim 5, wherein the re-scheduler is to receive a signal indicating that the cache miss corresponding to the association has been serviced.

9. (Original) The processor of claim 8, wherein the re-scheduler is to cause an instruction to be designated as eligible for re-execution based on the signal.

10. (Currently Amended) A method comprising:
generating a cache miss in a processor;

assigning an identifier to the cache miss and writing the identifier in a field of a load instruction generating the cache miss;

propagating the identifier to any instruction dependent on the load instruction;

issuing a request to service the cache miss to a memory system of the computer and providing the identifier to the memory system;

placing the load instruction in a queue for re-execution, where an eligibility of the instruction for re-execution is based at least in part on the identifier;

placing the instruction dependent on the load instruction in the queue for execution;

after the memory system completes servicing the request, causing the memory system to provide the identifier to the queue;

and designating the load instruction as eligible for re-execution based on the identifier provided by the memory system.

11. (Original) The method of claim 10, further comprising re-executing the load instruction based on receiving the identifier from the memory system.

12. (Canceled)

13. (Currently Amended) An apparatus comprising logic to:

enqueue a plurality of instructions needing re-execution due to respective cache misses in a re-execution queue;

associate each instruction in the queue with a respective corresponding cache miss;

propagate an association to a dependent instruction and enqueue the dependent instruction in the re-execution queue; and

after a cache miss is serviced, re-execute those instructions in the re-execution queue associated with the serviced cache miss.

14. (Previously Presented) The apparatus of claim 13, further comprising determining a priority of the instructions.

15. (Previously Presented) The apparatus of claim 13, wherein the associating comprises writing an identifier of a cache miss in an instruction.

16. (Currently Amended) A system comprising:

a memory system to hold instructions for execution; a processor coupled to the memory system, the processor including:

a re-scheduler to hold instructions from the memory system enqueued for execution;

association logic to form an association between a cache miss and ~~an~~ a first instruction generating the cache miss, the first instruction to be enqueued in the re-scheduler; and

propagation logic to propagate the association to ~~an~~ a second instruction dependent on the first instruction ~~generating the cache miss, the second instruction to be~~ enqueued in the re-scheduler.

17. (Original) The system of claim 16, wherein the re-scheduler is further coupled to priority logic to determine a priority of instructions in the re-scheduler.

18. (Original) The system of claim 16, wherein the association logic is to assign an identifier to the cache miss.